ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΡΓΑΣΤΗΡΙΟ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ Η/Υ

4ο Εξάμηνο

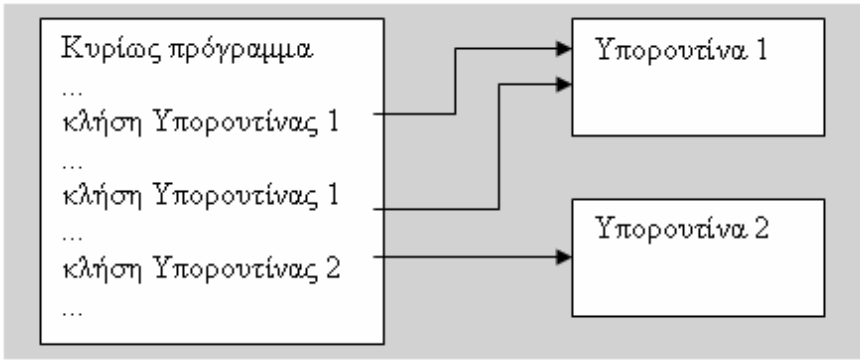
Μαδεμλής Ιωάννης

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΥΠΟΡΟΥΤΙΝΕΣ

Οι υπορουτίνες αποτελούν αυτόνομα τμήματα κώδικα που διεκπεραιώνουν μία συγκεκριμένη εργασία και μπορούμε να τα καλούμε μέσα από το κυρίως πρόγραμμα όσες φορές χρειαστεί

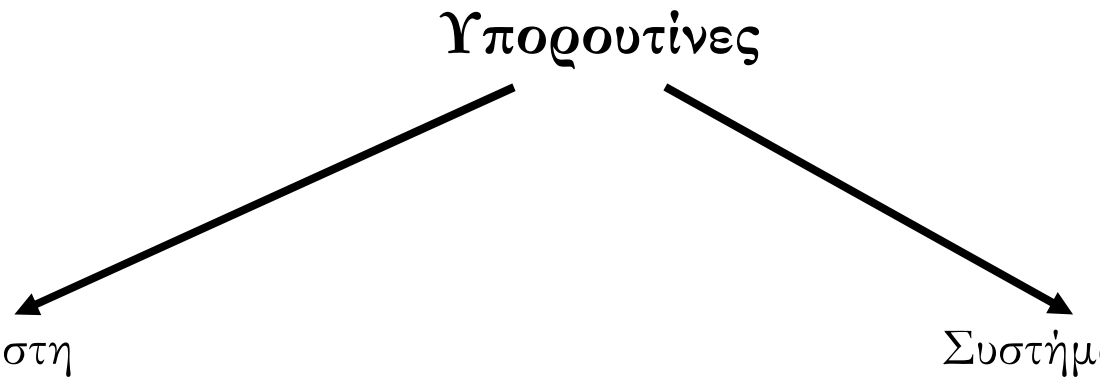
- Συμβάλλουν στην μείωση του συνολικού όγκου του κώδικα σε περίπτωση ύπαρξης όμοιων επαναλαμβανόμενων τμημάτων
- Καθιστούν την διόρθωση πιο εύκολη γιατί διορθώνουμε τον κώδικα της υπορουτίνας μία φορά και η διόρθωση ισχύει για όλο το πρόγραμμα
- Κάνουν τον κώδικα πιο κατανοητό και δίνουν την δυνατότητα επαναχρησιμοποίησης του κώδικα



ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΡΓΑΣΤΗΡΙΟ 6

ΥΠΟΡΟΥΤΙΝΕΣ ΣΤΗΝ **ASSEMBLY** ΤΟΥ **8088**



CALL (Κλήση υπορουτίνας χρήστη)

RET (Επιστροφή από υπορουτίνα)

INT (Κλήση υπορουτίνας συστήματος)

IRET (Επιστροφή από υπορουτίνα συστ.)

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΝΤΟΛΕΣ ΥΠΟΡΟΥΤΙΝΩΝ

Η εντολή CALL και οι τρόποι σύνταξής της

Κλήση υπορουτίνας χρήστη, CALL a

- CALL διεύθυνση μνήμης — π.χ. CALL 0500
- CALL καταχωρητής — π.χ. CALL DX
- CALL [διεύθυνση μνήμης] — π.χ. CALL [0500] ή CALL NE[0500]
- CALL segment:offset — π.χ. CALL 0100:0400
- CALL FAR [διεύθυνση μνήμης] — π.χ. CALL FAR[0600]

Η εντολή RET και οι τρόποι σύνταξής της

Επιστροφή από υπορουτίνα χρήστη στο κυρίως πρόγραμμα, RET
Είναι η τελευταία εντολή κάθε υπορουτίνας

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

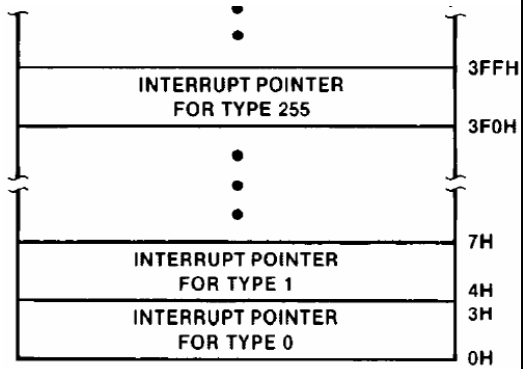
ΕΝΤΟΛΕΣ ΥΠΟΡΟΥΤΙΝΩΝ

- **Η εντολή INT και οι τρόποι σύνταξής της**

  Κλήση υπορουτίνας συστήματος (software interrupt), INT n

  - INT n (n → 0 ÷ FF) π.χ. INT 5

  Καλεί τη ρουτίνα συστήματος n. Η διεύθυνση της ρουτίνας αποτελείται από 4 bytes (2 bytes segment, 2 bytes offset) και είναι αποθηκευμένη στα πρώτα 1024 bytes της μνήμης (Interrupt Vectors, 00000_{16} …$003FF_{16}$), στη διεύθυνση n x 4 και στα επόμενα 3 bytes (low, high offset, low, high segment).

  - INTO

  Ελέγχει την τιμή της σημαίας υπερχείλισης και αν αυτή είναι 1 τότε εκτελεί την ρουτίνα διαχείρισης της υπερχείλισης (⇔ INT 4)

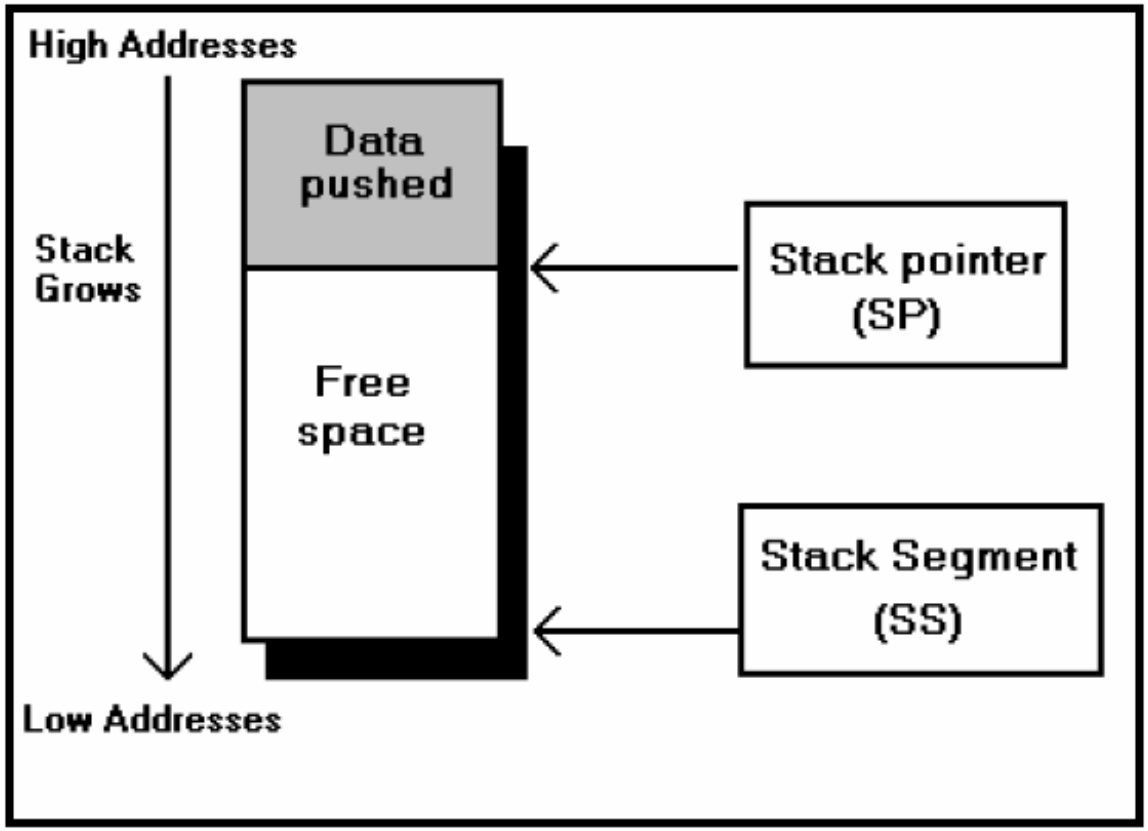
- **Η εντολή IRET και οι τρόποι σύνταξής της**

  Επιστροφή από υπορουτίνα συστήματος στο κυρίως πρόγραμμα, IRET

  Είναι η τελευταία εντολή κάθε υπορουτίνας συστήματος

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

Ο ΣΩΡΟΣ - STACK (ή ΣΤΟΙΒΑ)



Ο τρόπος λειτουργίας του σωρού stack.

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

Ο ΣΩΡΟΣ - STACK (ή ΣΤΟΙΒΑ)

Στο σωρό (δομή LIFO προσωρινής αποθήκευσης) αποθηκεύουν δεδομένα:

- Οι υπορουτίνες χρήστη με την εντολή CALL (διεύθυνση επιστροφής)
- Οι υπορουτίνες συστήματος με την εντολή INT (flag register, code segment and offset)
- Τα προγράμματα των χρηστών (οποιουσδήποτε καταχωρητές ή θέσεις μνήμης)

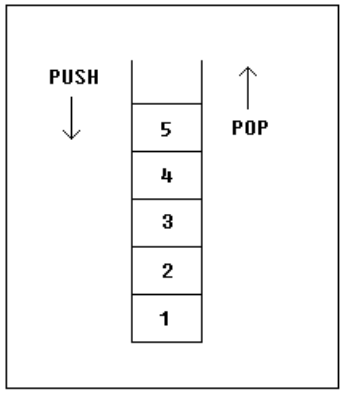
Η κορυφή του σωρού δίνεται από τη διεύθυνση SS:SP

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΝΤΟΛΕΣ ΔΙΑΧΕΙΡΙΣΗΣ ΣΩΡΟΥ

- Η εντολή PUSH/PUSHF και οι τρόποι σύνταξής της

  Εισάγει στο σωρό ένα 16 bit δεδομένο και ενημερώνει τον SP (SP=SP-2)
  - PUSH διεύθυνση μνήμης π.χ. PUSH [0500]
  - PUSH καταχωρητής π.χ. PUSH DX
  - PUSHF



- Η εντολή POP/POPF και οι τρόποι σύνταξής της

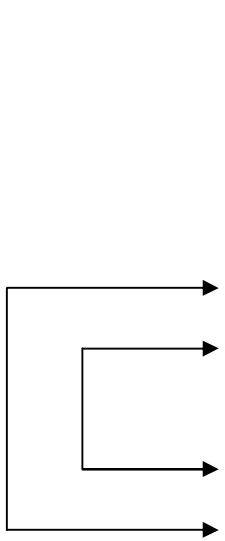
  Ανακαλεί από το σωρό ένα 16 bit δεδομένο και ενημερώνει τον SP (SP=SP+2)
  - POP διεύθυνση μνήμης π.χ. POP [0500]
  - POP καταχωρητής π.χ. POP DX
  - POPF

  Όσα PUSH έχει το πρόγραμμά μας <u>ακριβώς τόσα</u> πρέπει να είναι και τα POP

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

Ο ΣΩΡΟΣ - STACK (ή ΣΤΟΙΒΑ)

Παράδειγμα κώδικα αποθήκευσης και ανάκλησης καταχωρητών



```
...(προηγούμενες εντολές)
PUSH AX          ;σώσιμο του AX
PUSH BX          ;σώσιμο του BX
...
PUSH ES          ; σώσιμο του ES
PUSHF            ; σώσιμο των σημαιών
CALL XXXX
POPF             ; ανάκτηση των σημαιών
POP ES           ; ανάκτηση του ES
...
POP BX           ; ανάκτηση του BX
POP AX           ; ανάκτηση του AX
...(επόμενες εντολές)
```

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΡΓΑΣΤΗΡΙΟ 6

ΑΣΚΗΣΗ 6.1

Να γίνει πρόγραμμα που θα συγκρίνει τα περιεχόμενα δύο πινάκων 8 bit αριθμών που βρίσκονται στις θέσεις μνήμης **0200, 0201, 0202,….,0209** και **0300, 0301, 0302,…..0309** ανά δυο αντίστοιχα (0200 με 0300, 0201 με 0301, κ.ο.κ.) και να βάζει στις αντίστοιχες θέσεις του πίνακα **0400**, **0401**, **0402**,…,**0409** το **00** αν είναι ίσα και το **FF** αν είναι διαφορετικά.

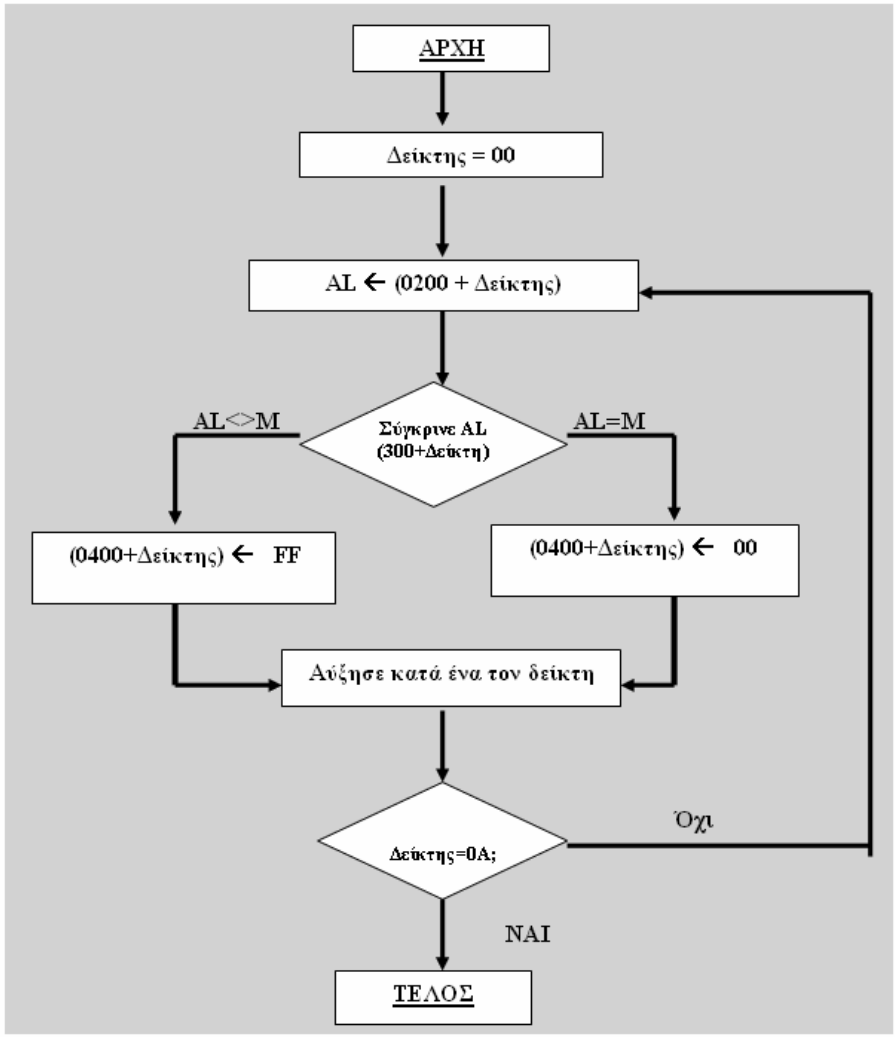
0200	FF
0201	23
0202	A0
0203	23
0204	45
0205	1A
0206	0C
0207	66
0208	99
0209	FC

0300	DD
0301	23
0302	A1
0303	32
0304	54
0305	1A
0306	AB
0307	AC
0308	AC
0309	FC

0400	FF
0401	**00**
0402	FF
0403	FF
0404	FF
0405	**00**
0406	FF
0407	FF
0408	FF
0409	**00**

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 6.1



ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 6.2

Να γραφεί πρόγραμμα που να προσθέτει δύο πίνακες 10 θέσεων που βρίσκονται στις διευθύνσεις μνήμης **0200** και **0300** αντίστοιχα και να τοποθετεί τον πίνακα-αποτέλεσμα στην θέση **0400**

0200	00		0300	00		0400	00
0201	01		0301	02		0401	03
0202	02		0302	03		0402	05
0203	03		0303	01		0403	04
0204	04	+	0304	06	=	0404	0A
0205	05		0305	07		0405	0C
0206	06		0306	05		0406	0B
0207	07		0307	00		0407	07
0208	08		0308	02		0408	0A
0209	09		0309	06		0409	0F

ΑΣΚΗΣΗ 6.3

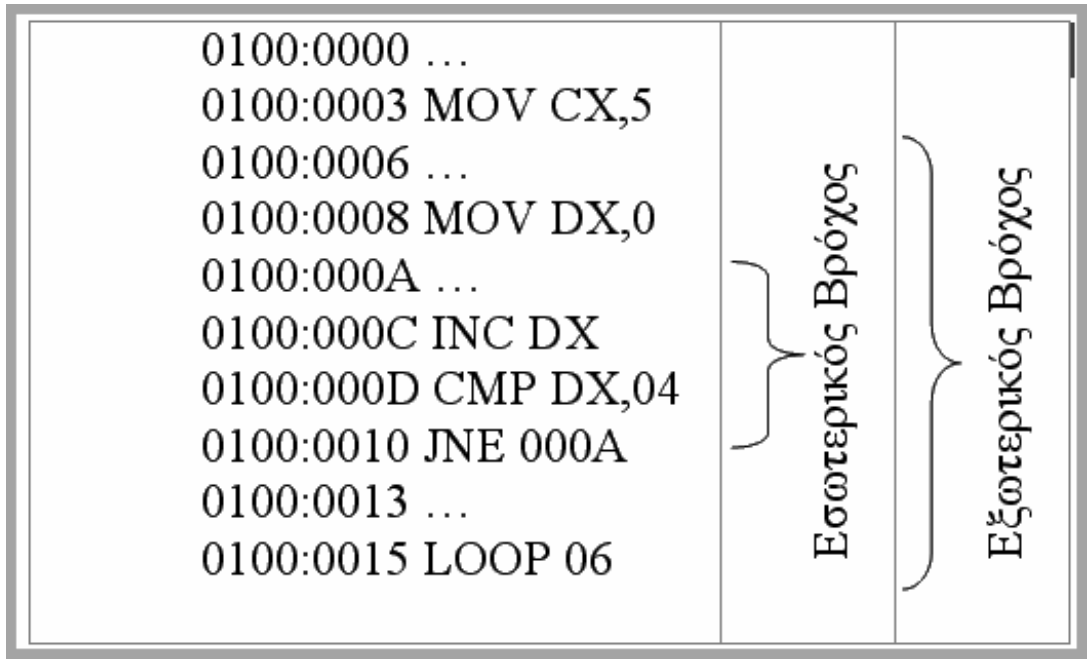
Να γραφεί το ίδιο πρόγραμμα με χρήση υπορουτίνας για την πρόσθεση κάθε στοιχείου των πινάκων

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΝΘΕΤΟΙ ΒΡΟΧΟΙ

Οι ένθετοι βρόχοι στην Assembly υλοποιούνται με κατάλληλες εντολές συνθήκης και αλμάτων υπό συνθήκη

Οι ένθετοι βρόχοι (που μπορεί να είναι και πάνω από 2 επιπέδων) χρησιμοποιούνται σε πλήθος αλγορίθμων, όπως η επεξεργασία διδιάστατων ή γενικά ν-διάστατων πινάκων, οι αλγόριθμοι ταξινόμησης, κ.λ.π.



ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΠΙΝΑΚΕΣ 2 ΔΙΑΣΤΑΣΕΩΝ

Οι πίνακες δύο διαστάσεων αποθηκεύονται σαν μονοδιάστατοι στην μνήμη και μπορούν να προσπελαστούν:

- Με απλό βρόχο όταν δεν μας ενδιαφέρει η θέση κάθε στοιχείου μέσα στον πίνακα
- Με διπλό βρόχο όταν μας ενδιαφέρει η θέση κάθε στοιχείου μέσα στον πίνακα

X_{11}	X_{12}	X_{13}
X_{21}	X_{22}	X_{23}
X_{31}	X_{32}	X_{33}

0200	0201	0202	0203	0204	0205	0206	0207	0208
X_{11}	X_{12}	X_{13}	X_{21}	X_{22}	X_{23}	X_{31}	X_{32}	X_{33}

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΠΙΝΑΚΕΣ 2 ΔΙΑΣΤΑΣΕΩΝ

- Προσπέλαση πίνακα 3X3 σειριακά με απλό βρόχο και ένα δείκτη

```
0100:0000    MOV SI,0000
0100:0003    MOV AL,[0200+SI]
0100:0006    …
0100:000A    INC SI
0100:000B    CMP SI, 9
0100:000E    JNE 0003
```

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΠΙΝΑΚΕΣ 2 ΔΙΑΣΤΑΣΕΩΝ

- Προσπέλαση (κατά γραμμές) πίνακα 3X3 με διπλό βρόχο, έναν δείκτη και 2 καταχωρητές παρακολούθησης γραμμής και στήλης

```
0100:0000   MOV SI,0000
0100:0003   MOV BL,00      ← καταχωρητής παρακολούθησης γραμμής
0100:0005   MOV CL,00      ← καταχωρητής παρακολούθησης στήλης
0100:0007   MOV AL, [0200+SI]
0100:000A   …
0100:0010   INC SI
0100:0011   INC CL         ← αύξηση στήλης
0100:0012   CMP CL,03      ← έλεγχος τέλους στηλών
0100:0015   JNE 007
0100:0018   MOV CL,00      ← μηδενισμός στήλης
0100:001B   INC BL         ← αύξηση γραμμής
0100:001C   CMP BL, 03     ← έλεγχος τέλους γραμμών
0100:001F   JNE 0007
```

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

Η ΕΝΤΟΛΗ XLAT

- Προσπέλαση Look-Up tables. Εντοπίζει ένα byte σε πίνακα στην μνήμη, την διεύθυνση βάσης του οποίου κρατά ο καταχωρητής BX (DS:BX) ενώ η θέση στον πίνακα (δείκτης) καθορίζεται από τον καταχωρητή AL. Το byte του πίνακα καταχωρείται πάλι στον AL

Π.χ.

```
MOV BX, 0200
MOV AL, 03
XLAT
```

AL ← 09 ($\Leftrightarrow 3^2$)

0200	00
0201	01
0202	04
0203	09
0204	10

Look-Up table τετραγώνου αριθμών 0-4

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΡΓΑΣΤΗΡΙΟ 7

ΑΣΚΗΣΗ 7.1

Να γίνει πρόγραμμα που να ταξινομεί κατ' αύξουσα σειρά, με τον αλγόριθμο Bubblesort, τους αριθμούς–τιμές 8 bit ενός πίνακα που βρίσκεται στις θέσεις μνήμης 0200–0209 (10 τιμές), δηλαδή το μικρότερο να τοποθετείται στη θέση 0200 και το μεγαλύτερο στη θέση 0209.

Πριν την ταξινόμηση (η χειρότερη περίπτωση ☹) :

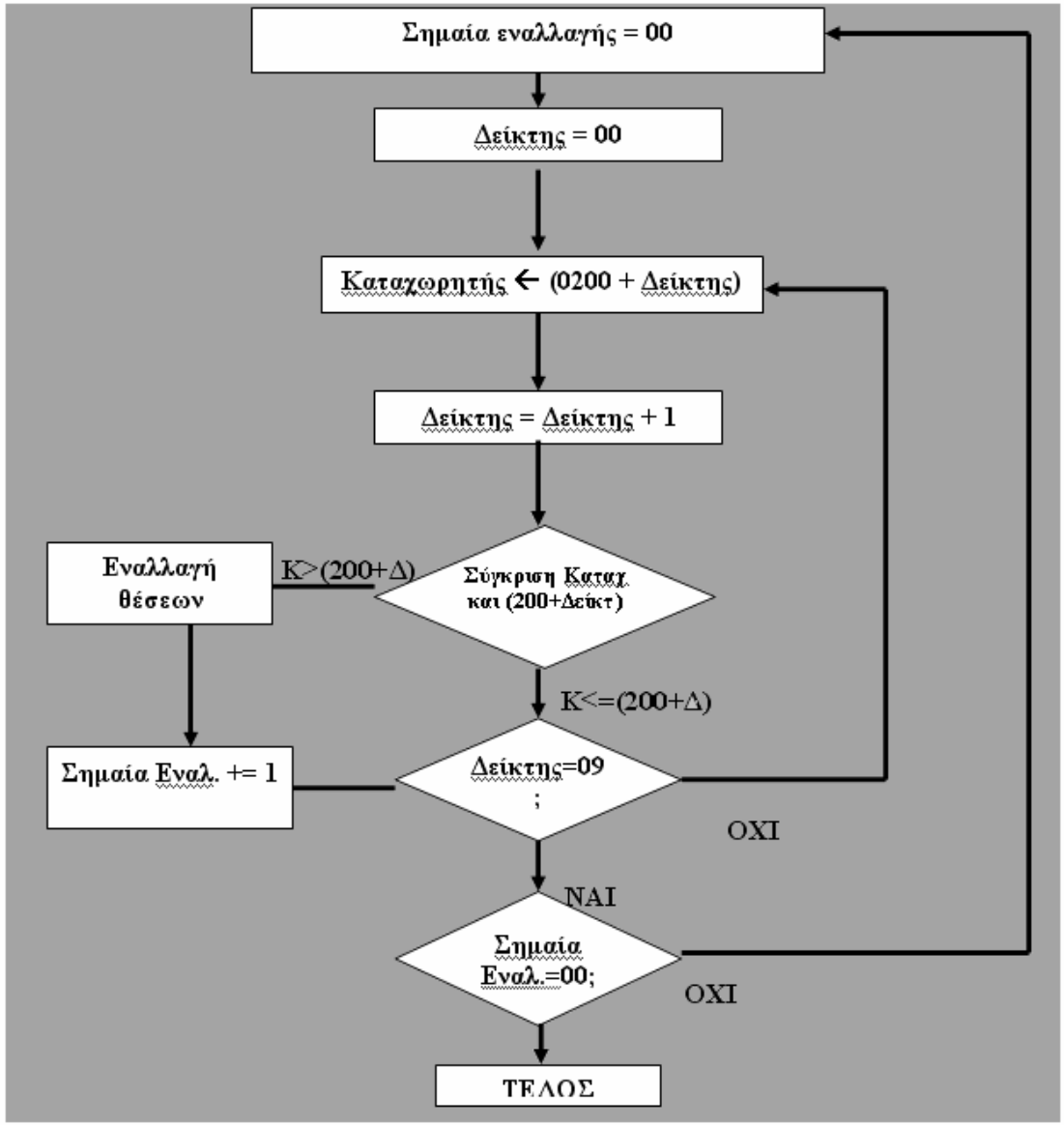
0200	**0201**	**0202**	**0203**	**0204**	**0205**	**0206**	**0207**	**0208**	**0209**
FF	FE	FD	FC	FB	FA	F9	F8	F7	F6

Μετά την ταξινόμηση ο πίνακας θα πρέπει να έχει τη μορφή :

0200	**0201**	**0202**	**0203**	**0204**	**0205**	**0206**	**0207**	**0208**	**0209**
F6	F7	F8	F9	FA	FB	FC	FD	FE	FF

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΕΡΓΑΣΤΗΡΙΟ 7



ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 7.2

Να αναπτυχθεί πρόγραμμα που να προσθέτει τα στοιχεία ενός διδιάστατου πίνακα 3X3 που βρίσκεται στην διεύθυνση μνήμης 0200 με χρήση διπλού ένθετου βρόχου. Η πρώτη γραμμή βρίσκεται στις θέσεις 0200, 0201, 0202, η δεύτερη γραμμή στις θέσεις 0203, 0204, 0205 και η τρίτη γραμμή στις θέσεις 0206, 0207, 208. Το αποτέλεσμα να αποθηκεύεται στη θέση 0209.

ΑΣΚΗΣΗ 7.3

Να αναπτυχθεί πρόγραμμα που να μας δίνει το τετράγωνο ενός αριθμού μέσω look-up table για τους αριθμούς από 1..100. Το look-up table θα κατασκευάζεται υπολογιστικά στην αρχή του προγράμματος και θα αρχίζει στην διεύθυνση 0400. Στη συνέχεια το πρόγραμμα θα λαμβάνει αριθμούς από έναν πίνακα 10 θέσεων που αρχίζει στην θέση 0200 και θα υπολογίζει μέσω look-up table τα τετράγωνα των αριθμών τα οποία και θα τοποθετεί σε πίνακα που θα αρχίσει στην θέση 0300.

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

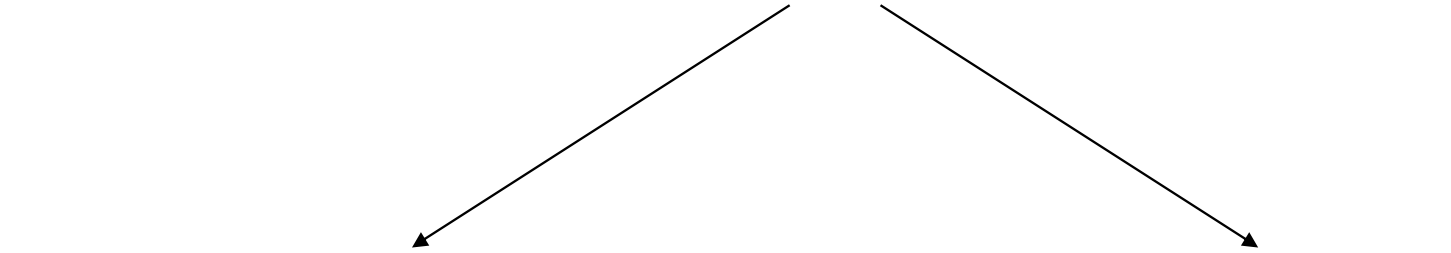
ΑΛΦΑΡΙΘΜΗΤΙΚΑ (STRINGS)

Τα αλφαριθμητικά είναι στην ουσία μονοδιάστατοι πίνακες αριθμών στην μνήμη (συνήθως του 1 byte), στους οποίους οι αριθμοί αντιστοιχούν σε χαρακτήρες μέσω της αντιστοίχισης του κώδικα ASCII. Τα αλφαριθμητικά συνήθως τερματίζουν με τον αριθμό 0 (χαρακτήρας '\0').

Παράδειγμα :

0200	0201	0202	0203	0204	0205
48_{16}	45_{16}	$4C_{16}$	$4C_{16}$	$4F_{16}$	00_{16}
H	E	L	L	O	

Ο χειρισμός των strings μπορεί να γίνει με



ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΛΦΑΡΙΘΜΗΤΙΚΑ (STRINGS)

- Η εντολή **REP**

  Επαναλαμβάνει την επόμενη εντολή αλφαριθμητικών τόσες φορές όσο η τιμή του CX, ο οποίος μειώνεται σε κάθε επανάληψη κατά 1, με τελική τιμή CX= 0. Λειτουργεί μόνο όταν η επόμενη εντολή είναι μία από τις εντολές αλφαριθμητικών MOVS, CMPS, SCAS, LODS, STOS.

  Παράδειγμα :

  ```
  MOV CX,5
  REP
  STOSB
  ```

  (Εκτελεί 5 φορές την εντολή STOSB)

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΛΦΑΡΙΘΜΗΤΙΚΑ (STRINGS)

Η εντολή **MOVSB & MOVSW**

(Move String Byte/Word – Αντιγραφή byte/word από String σε String)

Αντιγράφει ένα byte/word ενός string από την διεύθυνση DS:SI στην διεύθυνση ES:DI. Αμέσως μετά αυξάνει (DF=0) ή μειώνει (DF=1) τους καταχωρητές SI και DI κατά 1 (2 για word). Το Direction flag επηρεάζεται από τις εντολές CLD (DF=0) και STD (DF=1)

Παράδειγμα :

```
MOV AX,0100
MOV DS,AX
MOV ES,AX
MOV SI, 200
MOV DI, 300
MOV CX, 4
REP MOVSB          ; DS:SI → ES:DI
```

(Αντιγράφει 4 bytes από το string στην θέση **DS:SI** (0100:0200) στο string που είναι στη θέση **ES:DI** (0100:0300), αυξάνοντας τους SI, DI σε κάθε επανάληψη και μειώνοντας τον CX)

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΛΦΑΡΙΘΜΗΤΙΚΑ (STRINGS)

Η εντολή **LODSB & LODSW**

(String Byte/Word – Φόρτωσε byte/word ενός string)

Φορτώνει στον καταχωρητή AL ή AX (byte/word) το περιεχόμενο της θέσης μνήμης που είναι αποθηκευμένη στους καταχωρητές DS:SI. Αμέσως μετά αυξάνει ή μειώνει τον καταχωρητή SI κατά 1 (2 για word), ανάλογα με την τιμή της σημαίας κατεύθυνσης (DF).

Παράδειγμα :

```
MOV AX,0100
MOV DS,AX
MOV SI,0500
LODSB                ; DS:SI → AL
MOV [200],AL
```

(Φορτώνει 1 byte από τη θέση **DS:SI** (0100:0500) στον AL και από τον AL στη θέση μνήμης 0200)

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΛΦΑΡΙΘΜΗΤΙΚΑ (STRINGS)

Η εντολή **STOSB & STOSW**

(Store String Byte/Word – Αποθήκευσε byte/word σε string)

Γράφει το περιεχόμενο του καταχωρητή AL ή AX (1 ή 2 byte) στην θέση μνήμης που είναι αποθηκευμένη στους καταχωρητές ES:DI. Αμέσως μετά αυξάνει ή μειώνει τον καταχωρητή DI κατά 1, ανάλογα με την τιμή της σημαίας κατεύθυνσης (DF).

Παράδειγμα :

```
MOV BX,0100
MOV ES,BX
MOV DI,0600
MOV AL,41
MOV CX,5
REP STOSB                ; AL → ES:DI
```

(Αποθηκεύει 5 bytes 41_{16} ή 'A' στην θέση **ES:DI** (0100:0600) , αυξάνοντας τον DI σε κάθε επανάληψη ώστε τα 'A' να αποθηκεύονται σε διαδοχικές θέσεις)

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΛΦΑΡΙΘΜΗΤΙΚΑ (STRINGS)

Η εντολή **CMPSB & CMPSW**

(Compare String Byte/Word – Σύγκρινε byte/word από δύο string)

Συγκρίνει ένα byte (ή 2) ενός string από την διεύθυνση DS:SI με ένα byte (ή 2) ενός άλλου string στην διεύθυνση ES:DI, επηρεάζοντας τις σημαίες (flags). Αμέσως μετά αυξάνει ή μειώνει τους καταχωρητές SI και DI κατά 1 (2 για word), ανάλογα με την τιμή της σημαίας κατεύθυνσης (DF).

Παράδειγμα:

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΛΦΑΡΙΘΜΗΤΙΚΑ (STRINGS)

- Η εντολή **SCASB & SCASW**

(Scan String Byte/Word – Αναζήτηση byte/word μέσα σε string)

Συγκρίνει το περιεχόμενο του καταχωρητή AL (AX για word) με ένα byte (2 για word) ενός string στην διεύθυνση ES:DI, επηρεάζοντας τις σημαίες (flags). Αμέσως μετά αυξάνει ή μειώνει τον καταχωρητή DI κατά 1 (2 για word), ανάλογα με την τιμή της σημαίας κατεύθυνσης (DF).

Παράδειγμα :

```
MOV BX,0100
MOV ES,BX
MOV AL, 41
MOV DI, 300
SCASB
```

(Συγκρίνει τον AL με 1 byte από το string στην θέση **ES:DI** (0100:0300) , επηρεάζοντας τις σημαίες, και αυξάνοντας τον DI)

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 8.1

Να γίνει πρόγραμμα που θα ψάχνει εάν σ' έναν πίνακα που βρίσκεται στις θέσεις **0200-0209** υπάρχει η τιμή που βρίσκεται στη θέση **020A** και θα σημειώνει στις θέσεις **0300** και πέρα σε ποια θέση (1η ,2η , 3η , ….) από τις 10 (0200-0209) βρίσκεται αυτή. Π.χ. εάν η ζητούμενη τιμή βρίσκεται στις θέσεις **0203**, **0207** τότε το πρόγραμμα θα βάλει στις θέσεις **0300**, **0301** τις τιμές:

0300: 04 (4ο στοιχείο του πίνακα)

0301: 08 (8ο στοιχείο του πίνακα)

Το πρόγραμμα να υλοποιηθεί μία φορά με δεικτοδοτούμενη διευθυνσιοδότηση και μία φορά με εντολές αλφαριθμητικών.

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 8.1

Για παράδειγμα:

020A

FF	**Τιμή αναζήτησης**

0200	**0201**	**0202**	**0203**	**0204**	**0205**	**0206**	**0207**	**0208**	**0209**
FF	66	FD	78	**FB**	**FF**	**FF**	F8	F7	F6

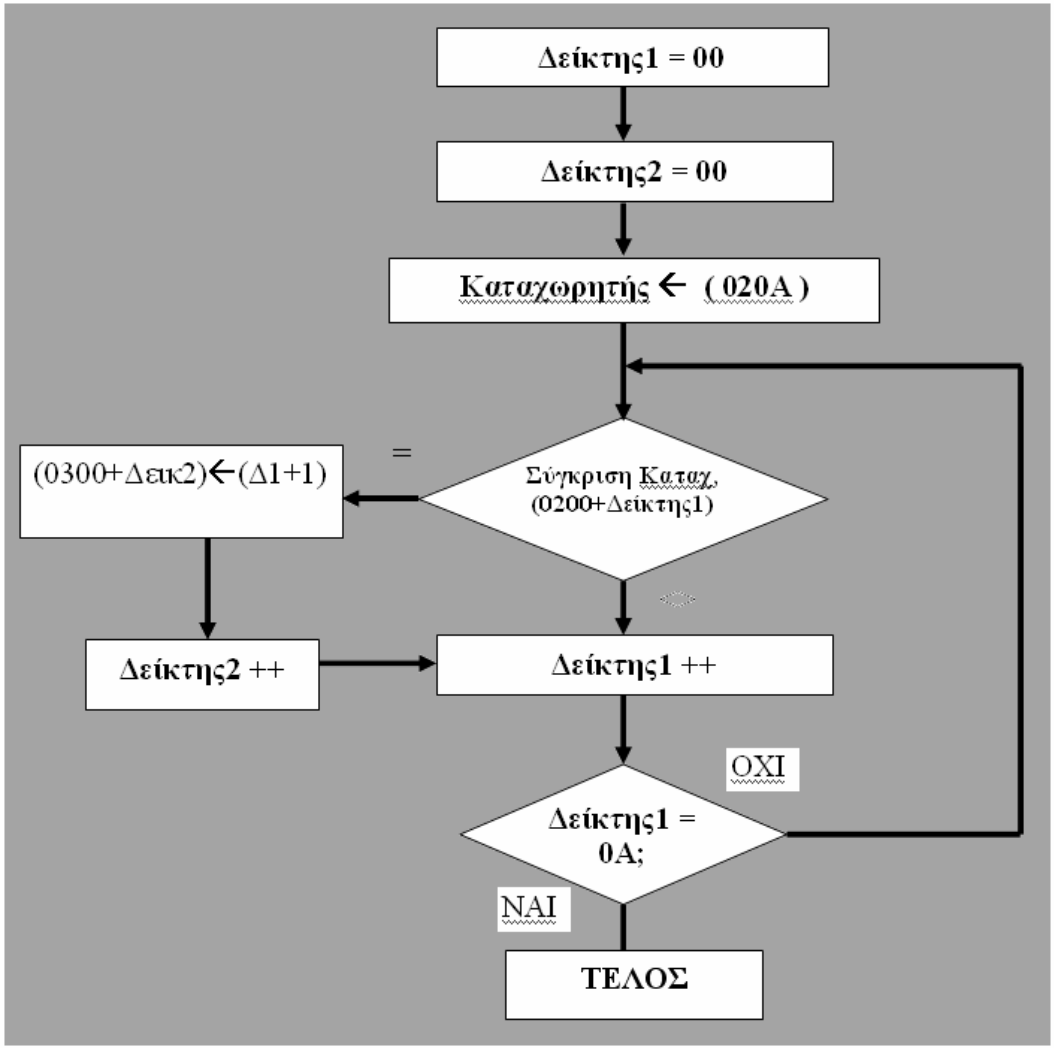
Αποτέλεσμα :

0300	**0301**	**0302**	**0303**	**0304**	**0305**	**0306**	**0307**	**0308**	**0309**
01	**06**	**07**							

Υπόδειξη : Πριν εκτελέσετε το πρόγραμμα, με τη βοήθεια του monitor, γεμίστε τις διευθύνσεις μνήμης **0300..0309** με την τιμή **00**, ώστε να μην επηρεαστεί το αποτέλεσμα από προηγούμενες τιμές. (Εντολή F 300,309,00)

ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 8.1



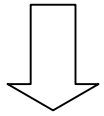
ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 8.2

Να αναπτυχθεί πρόγραμμα που θα αντιγράφει ένα string με μέγιστο μήκος 8 χαρακτήρες, από τη θέση **0200** στη θέση **0208**. Θεωρείστε ότι το τέλος του string σηματοδοτείται με το '\0' (αριθμός 0 όχι χαρακτήρας '0'). Η υλοποίηση να γίνει μία φορά με την εντολή MOVS και 1 φορά με τις εντολές LODSB, STOSB καθώς και με δεικτοδοτούμενη προσπέλαση.

Μπορεί να γίνει δοκιμή με το string :

0200	0201	0202	0203	0204	0205
T	**O**	**D**	**A**	**Y**	
54_{16}	$4F_{16}$	44_{16}	41_{16}	59_{16}	00_{16}



0208	0209	020A	020B	020C	020D
T	**O**	**D**	**A**	**Y**	
54_{16}	$4F_{16}$	44_{16}	41_{16}	59_{16}	00_{16}

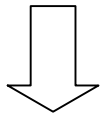
ΤΕΙ ΣΕΡΡΩΝ
Τμήμα Πληροφορικής & Επικοινωνιών
Τομέας Αρχιτεκτονικής Υπολογιστών & Βιομηχανικών Εφαρμογών

ΑΣΚΗΣΗ 8.3

Να αναπτυχθεί πρόγραμμα που θα μετατρέπει ένα string από κεφαλαία σε πεζά. Θεωρείστε ότι το αρχικό string με τα κεφαλαία βρίσκεται στη θέση μνήμης **0200** και έχει μέγιστο μήκος 8 χαρακτήρες ενώ το τέλος του string σηματοδοτείται με το '\0' (αριθμός 0 όχι χαρακτήρας '0'). Το τελικό string να αποθηκεύεται στη διεύθυνση μνήμης **0208**.

Μπορεί να γίνει δοκιμή με το string :

0200	**0201**	**0202**	**0203**	**0204**	**0205**
H	**E**	**L**	**L**	**O**	
48_{16}	45_{16}	$4C_{16}$	$4C_{16}$	$4F_{16}$	00_{16}



0208	**0209**	**020A**	**020B**	**020C**	**020D**
h	**e**	**l**	**l**	**o**	
68_{16}	65_{16}	$6C_{16}$	$6C_{16}$	$6F_{16}$	00_{16}